Safety Verification of Communicating One-Counter Machines

Alexander Heußner[1], Tristan Le Gall[2], and Grégoire Sutre[3]

1 **ULB, Brussels, Belgium & University of Bamberg, Bamberg, Germany**
2 **CEA, LIST, DILS/LMeASI, Gif-sur-Yvette, France**
3 **Univ. Bordeaux & CNRS, LaBRI, UMR 5800, Talence, France**

Abstract

In order to verify protocols that tag messages with integer values, we investigate the decidability of the reachability problem for systems of communicating one-counter machines. These systems consist of local one-counter machines that asynchronously communicate by exchanging the value of their counters via, a priori unbounded, Fifo channels. This model extends communicating finite-state machines (Cfsm) by infinite-state local processes and an infinite message alphabet. The main result of the paper is a complete characterization of the communication topologies that have a solvable reachability question. As already Cfsm exclude the possibility of automatic verification in presence of mutual communication, we also consider an under-approximative approach to the reachability problem, based on rendezvous synchronization.

1998 ACM Subject Classification F.1.1 Models of Computation, D.2.4 Program Verification

Keywords and phrases Counter Machines, Fifo Channels, Reachability Problem, Data Words

Digital Object Identifier 10.4230/LIPIcs.FSTTCS.2012.224

1 Introduction

One of the most challenging and imperative problems in computer science today is the verification of the nowadays ubiquitous distributed systems, as these are increasingly applied in vital and sensitive areas. Such systems consist of several processes that asynchronously exchange data over a network topology. A well-established model, known as *communicating finite-state machines* (Cfsm), combines local finite-state machines with point-to-point, unbounded Fifo queues that pass messages from a finite alphabet. Cfsm laid the foundation for a family of infinite-state models parametrized by the computational power of the local machines, such as communicating Petri nets [10] and pushdown systems [14, 13].

However, basic safety verification questions, like reachability, are known to be undecidable for Cfsm already on simple topologies [6, 17]. One important line of current research is the influence of the underlying communication topology to these verification questions when we restrict the interplay between communication and the local machine's power [14, 7, 13]. In this paper, we extend this research towards the verification of communicating machines that locally use counters and can exchange these via message passing, thus introducing two additional sources of infinity to Cfsm's unbounded channels. Infinite message alphabets are demanded in practice to model protocols based on (*a priori* unbounded) sequence numbers.

Motivating Example. A simple sliding window protocol is depicted in Figure 1. A *sender* transmits a sequence number (ignoring additional data) to a *receiver* that advances the expected sequence number if it got the right message, demands to resend the expected

32nd Int'l Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2012).
Editors: D. D'Souza, J. Radhakrishnan, and K. Telikepalli; pp. 224–235

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

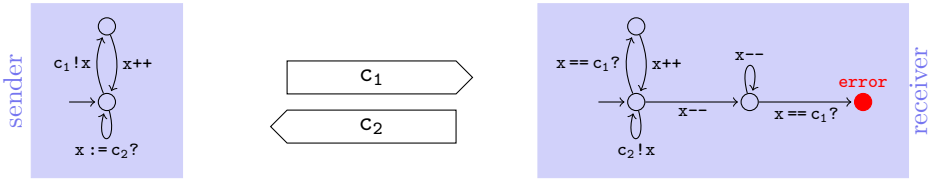

Figure 1 A simple sliding window protocol: sender on the left, receiver on the right.

message, or fails if the sequence number was already received. Checking the correctness of such protocols (here, whether the `error` state is reachable) is the main topic of this paper.

Contributions. We present the formal model of *systems of communicating one-counter machines*. This model is parametrized by a communication topology, specifying point-to-point Fifo channels between processes. Processes are one-counter machines that can send or receive the contents of their local counter. We consider an extension of one-counter machines where tests are not limited to zero-tests $x = 0$, but can be any unary Presburger predicate $\varphi(x)$. Channels are a priori unbounded, and messages are natural numbers. Different ways of relating these messages to the machine's local counters are investigated. As our main result, we establish a complete classification of the topologies over which the reachability problem for systems of communicating one-counter machines is decidable. The underlying proof relies, on the one hand, on a reduction from the well-known undecidability of the reachability problem for two-counters Minsky machines. On the other hand, we use a reduction technique that inductively combines one-counter machines along a hierarchical order, which is based on the topology. This way, the reachability problem is reduced to the case of two processes that are connected by one channel. We show that the reachability problem is decidable in this setting.

Our decidability results are based on summarizing the behavior of a process between each communication action. Recall that the reachability relation of a one-counter machine is definable in Presburger arithmetic (see, e.g., [11]). But Presburger-definable binary relations are not closed under transitive closure, which makes them unsuitable for our summarization-based approach. As key ingredient to our proofs, we exhibit a class of binary Presburger predicates that corresponds exactly to one-counter reachability relations. Our characterization entails that this class is effectively closed under transitive closure, and that one-counter reachability relations are effectively closed under intersection.

As the undecidable topologies include cyclic architectures, that nevertheless are important in practice to permit mutual communication, we also consider an under-approximative approach based on *eager* runs, i.e., runs where a send action is directly followed by its reception. We characterize the strongly-connected topologies that have a decidable eager-reachability problem. In particular, the topology of our motivating example, which is a cycle of length two, allows to decide the verification problem (for eager runs).

Related Works. The basic undecidability result for Cfsm [6] is the corner stone for most ongoing research on models based local machines that communicate over Fifo channels. Prominent approaches to regain decidability for reachability/safety are restrictions on the size of the channels or the message alphabets (already in [6, 17]), as well as the focus on lossy channel systems [9, 1]. Recent research dealt with the influence of the underlying topology on decidability questions, e.g., systems mixing lossy and perfect channels [7]. Communicating pushdown machines focus on a typing of channel ends that forces the decoupling of pushdown and channel actions [14, 13]. Restricting the local pushdown alphabet to a singleton, but

extending the finite message alphabet to an infinite one leads in our case to an incomparable model. However, we similarly arrive at favorable decidability results for tree-like architectures, which are more restricted than those in [13] even when regarding only eager communication.

CFSM-style systems with *infinite* message alphabets were discussed in [15], but this work focused on the definition of a static analysis technique, and thus the practical implementation of verification algorithms. Also closely related are data words and their different underlying automata models that rely on an infinite input/output alphabet and local registers [3, 4]. However, these automata only allow to use an equality test on the infinite data alphabet and not to modify and test registers like counters do.

Counter machines are a classical formalism in computer science [16]. Besides the *two-counters* (Minsky) machines, which are Turing-complete, the verification of *one-counter* automata has gained a renewed interest recently [8, 12, 2]. Using one-counter automata with Presburger tests also appears in [5], yet only as symbolic representation of reachability sets and not as operational model for the underlying programs.

Outline. We introduce systems of communicating one-counter machines in Section 2. Section 3 presents our main result: the characterization of communication topologies that have a solvable reachability question. The proof of the positive case is provided in Section 4. Section 5 presents preliminary results on the decidability of the reachability question when we only consider eager runs. Some conclusions and perspectives are given in Section 6.

2 Systems of Communicating One-Counter Machines

Given a (possibly infinite) alphabet M, let M^* denote the set of all finite *words* over M, $\varepsilon \in M^*$ the *empty* word, and $u \cdot v$ the *concatenation* of two words $u, v \in M^*$. For a set of values X and a finite set of indices I, we write X^I for the set of all mappings from I to X. Such mappings may be interpreted as I-indexed X-valued *vectors*. Let x^i denote the i-th component of a vector $\boldsymbol{x} \in X^I$. Two constant vectors are introduced, for convenience: $\mathbf{0} \in \mathbb{N}^I$, which maps every index to 0, and $\boldsymbol{\varepsilon} \in (M^*)^I$, which maps every index to ε.

Communication Topologies. In our framework, channels are point-to-point. Each channel c has a source endpoint $\mathsf{src}(c)$, and a destination endpoint $\mathsf{dst}(c)$. These endpoints are pairs $(p, *)$ where p is the process communicating at the endpoint, and $* \in \{\bullet, \circ\}$ is the communication type of the endpoint. We introduce the types $\bullet$ and $\circ$ to model two communication policies that relate the message and the local counter of a machine before and after communication on an endpoint. We assert that $\circ$ is more restrictive than $\bullet$, namely, that the value of the local counter is "lost" by a communication with type $\circ$. This difference is formalized in the semantics introduced subsequently. First, let us formally define communication topologies.

▶ **Definition 2.1.** A *topology* is a quadruple $\mathcal{T} = \langle P, C, \mathsf{src}, \mathsf{dst} \rangle$ where P is a finite, non-empty set of *processes*, C is a finite, possibly empty set of *channels*, $\mathsf{src} : C \to P \times \{\bullet, \circ\}$ is a *source* mapping, and $\mathsf{dst} : C \to P \times \{\bullet, \circ\}$ is a *destination* mapping.

For better readability, we slightly abuse notation by identifying an endpoint $(p, *)$ with its process p or its type $*$, depending on the context. For instance, we write $\mathsf{src}(c) = p$ instead of $\mathsf{src}(c) = (p, *)$ for some $* \in \{\bullet, \circ\}$. Given a process $p \in P$, we let $C(p)$ denote the set of all channels with source or destination p. Formally, $C(p) = \{c \in C \mid \mathsf{src}(c) = p \vee \mathsf{dst}(c) = p\}$. The communication *type* of a process p on a channel $c \in C(p)$ that is not a self-loop, written $\mathsf{typ}(p, c)$, is the unique $* \in \{\bullet, \circ\}$ such that $(p, *)$ is an endpoint of c.

For each channel $c \in C$, we let $\overset{c}{\rightarrowtail}$ denote the binary relation on the set of processes P defined by $p \overset{c}{\rightarrowtail} q$ if $p = \mathsf{src}(c)$ and $q = \mathsf{dst}(c)$. Naturally, any topology may be viewed as the labeled *directed* graph $(P, \{\overset{c}{\rightarrowtail}\}_{c \in C})$. We assume some familiarity with classical notions on directed graphs, such as weak connectedness, strong connectedness, leaf nodes, etc. We also introduce the undirected binary relation $\overset{c}{\leftrightarrow}$, defined by $p \overset{c}{\leftrightarrow} q$ if $p \overset{c}{\rightarrowtail} q$ or $p \overset{c}{\leftarrowtail} q$. An *undirected path* in $\mathcal{T}$ is an alternating sequence $(p_0, c_1, p_1, \ldots, c_n, p_n)$, of processes $p_i \in P$ and channels $c_i \in C$, such that $p_{i-1} \overset{c_i}{\leftrightarrow} p_i$ for all $i \in \{1, \ldots, n\}$. Moreover, the undirected path is called *simple* when $p_0, \ldots, p_n$ are distinct. A *simple undirected cycle* in $\mathcal{T}$ is an undirected path $(p_0, c_1, p_1, \ldots, c_n, p_n)$, with $n \geq 1$, such that $p_1, \ldots, p_n$ are distinct, $c_1, \ldots, c_n$ are distinct, and $p_0 = p_n$. A *simple undirected shunt* in $\mathcal{T}$ is a simple undirected path $(p_0, c_1, p_1, \ldots, c_n, p_n)$, with $n \geq 2$, such that $\mathsf{typ}(p_0, c_1) = \bullet$ and $\mathsf{typ}(p_n, c_n) = \bullet$.

▶ **Definition 2.2.** Let $\mathcal{T}$ be a topology. $\mathcal{T}$ is called *cycle-free* if it contains no simple undirected cycle. $\mathcal{T}$ is called *shunt-free* if it contains no simple undirected shunt.

▶ Remark. Our notion of shunt is close to the *confluence* criterion presented in [13] for communicating pushdown processes. Simply put, confluence permits to synchronize two pushdown stacks, and a shunt permits to synchronize two counters, as will be seen later. However, shunts require at least one additional, intermediary process whereas confluence can be established directly between two processes. In our case, the topology $\mathtt{p} \overset{c}{\rightarrowtail} \mathtt{q}$ with channel endpoints of type $\bullet$ is shunt-free, and will be shown to have a decidable reachability problem.

Systems of Communicating One-Counter Machines. Classically, one-counter machines are finite-state automata, equipped with a counter, represented by a variable $\mathtt{x}$, that holds a non-negative integer value. The counter is initially set to zero, and can be incremented, decremented (provided that it remains non-negative), and tested for zero. In this paper, we consider an extension of counter machines where tests can be any unary Presburger predicate $\varphi(\mathtt{x})$. Such Presburger tests do not increase the expressive power of one-counter machines in terms of recognized languages [5]. We will show in the next section that the same property holds for their binary reachability relations. Presburger tests will be handy to merge several communicating one-counter machines in a single communicating one-counter machine.

Recall that *Presbuger arithmetic* is the first-order theory of the natural numbers with addition. A n-ary *Presburger predicate* is a Presburger formula φ with exactly n free variables. As usual, we write $\varphi(\mathtt{x}_1, \ldots, \mathtt{x}_n)$ to indicate that $\mathtt{x}_1, \ldots, \mathtt{x}_n$ are the free variables of φ. We let $\mathcal{P}_n$ denote the set of all n-ary Presburger predicates.

▶ **Definition 2.3.** A *system of communicating one-counter machines* is a pair $\mathcal{S} = \langle \mathcal{T}, (\mathcal{M}^p)_{p \in P} \rangle$ where $\mathcal{T}$ is a topology and, for each process p in P, $\mathcal{M}^p$ is a quintuple $\mathcal{M}^p = \langle S^p, I^p, F^p, A^p, \Delta^p \rangle$, called a *communicating one-counter machine*, where

- S^p is a finite set of *states*,
- $I^p, F^p \subseteq S^p$ are subsets of *initial states* and *final states*,
- $A^p \subseteq A_{\mathrm{cnt}} \cup A^p_{\mathrm{com}}$ is a finite set of *actions*, where
$$A_{\mathrm{cnt}} = \{\mathtt{add}(k) \mid k \in \mathbb{Z}\} \cup \{\mathtt{test}(\varphi) \mid \varphi \in \mathcal{P}_1\}$$
$$A^p_{\mathrm{com}} = \{c! \mid c \in C \wedge \mathsf{src}(c) = p\} \cup \{c? \mid c \in C \wedge \mathsf{dst}(c) = p\}$$
- $\Delta^p \subseteq S^p \times A^p \times S^p$ is a finite set of transition *rules*.

We give the operational semantics $[\![\mathcal{S}]\!]$ of a system of communicating one-counter machines $\mathcal{S}$ as a labeled transition system. A *configuration* of $[\![\mathcal{S}]\!]$ is triple $\sigma = (\boldsymbol{s}, \boldsymbol{x}, \boldsymbol{w})$ where $\boldsymbol{s}$ maps each process p to a state in S^p, $\boldsymbol{x}$ maps each process p to a counter value in $\mathbb{N}$, and $\boldsymbol{w}$ maps each channel c to a word over the set of natural numbers. Formally, the set of

configurations of $[\![\mathcal{S}]\!]$ is $(\prod_{p\in P} S^p) \times \mathbb{N}^P \times (\mathbb{N}^*)^C$. An *initial configuration* is a configuration $(\boldsymbol{s}, \boldsymbol{x}, \boldsymbol{w})$ such that $\boldsymbol{x} = \mathbf{0}$, $\boldsymbol{w} = \boldsymbol{\varepsilon}$, and $s^p \in I^p$ for all $p \in P$. Analogously, a *final configuration* is a configuration $(\boldsymbol{s}, \boldsymbol{x}, \boldsymbol{w})$ such that $\boldsymbol{x} = \mathbf{0}$, $\boldsymbol{w} = \boldsymbol{\varepsilon}$, and $s^p \in F^p$ for all $p \in P$. The *transition relation* of $[\![\mathcal{S}]\!]$, written $\rightarrow$, is the set of all triples (σ_1, a, σ_2), where $\sigma_1 = (\boldsymbol{s}_1, \boldsymbol{x}_1, \boldsymbol{w}_1)$ and $\sigma_2 = (\boldsymbol{s}_2, \boldsymbol{x}_2, \boldsymbol{w}_2)$ are configurations, and a is an action in A^p, for some $p \in P$, satisfying the following conditions:

- $(s_1^p, a, s_2^p) \in \Delta^p$ and $s_1^q = s_2^q$ for all $q \in P$ with $q \neq p$,
- if $a = \mathtt{add}(k)$ then $x_2^p = x_1^p + k$, $x_1^q = x_2^q$ for all $q \in P$ with $q \neq p$, and $\boldsymbol{w}_1 = \boldsymbol{w}_2$,
- if $a = \mathtt{test}(\varphi(\mathtt{x}))$ then the valuation $\{\mathtt{x} \mapsto x_1^p\}$ satisfies $\varphi(\mathtt{x})$, $\boldsymbol{x}_1 = \boldsymbol{x}_2$ and $\boldsymbol{w}_1 = \boldsymbol{w}_2$,
- if $a = c!$, then
  - $w_2^c = w_1^c \cdot x_1^p$ and $w_1^d = w_2^d$ for all $d \in C$ with $d \neq c$, and
  - if $\mathsf{src}(c) = \bullet$ then $\boldsymbol{x}_1 = \boldsymbol{x}_2$; otherwise $x_1^q = x_2^q$ for all $q \in P$ with $q \neq p$.
- if $a = c?$, then
  - $w_1^c = x_2^p \cdot w_2^c$ and $w_1^d = w_2^d$ for all $d \in C$ with $d \neq c$, and
  - if $\mathsf{dst}(c) = \bullet$ then $\boldsymbol{x}_1 = \boldsymbol{x}_2$; otherwise $x_1^q = x_2^q$ for all $q \in P$ with $q \neq p$.

For readability, we write $\sigma_1 \xrightarrow{a} \sigma_2$ in place of $(\sigma_1, a, \sigma_2) \in \rightarrow$. Notice that we do not explicitly index actions by the process that fires them, but we assert that one implicitly knows which process moves on each transition. A *run* of $[\![\mathcal{S}]\!]$ is a finite, alternating sequence $\rho = (\sigma_0, a_1, \sigma_1, \ldots, a_n, \sigma_n)$ of configurations σ_i and actions a_i, satisfying $\sigma_{i-1} \xrightarrow{a_i} \sigma_i$ for all i. We say that ρ is a run *from* σ_0 *to* σ_n, and, abusing notation, we shortly write $\rho = \sigma_0 \xrightarrow{*} \sigma_n$. The *length* of ρ is n, and is denoted by $|\rho|$. A run of length zero consists of a single configuration. A *full run* of $[\![\mathcal{S}]\!]$ is a run from an initial configuration to a final configuration.

The semantics of counter operations $\mathtt{add}(k)$ and $\mathtt{test}(\varphi)$ is the usual one. A send or receive action on a channel appends or removes a message in $\mathbb{N}$, as one would expect. However, there are additional restrictions on the interplay of the communicated message and the local counter. If the endpoint of the channel has type $\bullet$, the message must equal the value of the counter *before* and *after* the action. So the value of the counter is not modified by a communication on this endpoint. On the contrary, if the endpoint has type $\circ$, then the local counter value is "lost" by a communication on this endpoint:

- an emission transfers the value of the counter from the process to the channel; the counter is non-deterministically set to an arbitrary value after the emission.
- a reception transfers the message from the channel to the local counter; the behavior mirrors that of an emission.

Exchange of Messages from a Finite Alphabet. On the contrary to classical *communicating finite-state machines* (CFSM), communicating one-counter machines cannot (directly) send or receive messages from an arbitrary finite alphabet M. However, they are able to perform these actions indirectly, as follows. Assume, without loss of generality, that M is a finite set of natural numbers. Sending a message $m \in M$ on a channel c, like a CFSM would, simply amounts to setting the local counter to m, and performing an emission on c. Receiving a message $m \in M$ from a channel c, like a CFSM would, is done by performing a reception from c, and checking that the received message is m. To realize this check, the machine

- simply sets its counter to m before the reception, for an endpoint with type $\bullet$,
- or checks that the counter equals m after the reception, for an endpoint with type $\circ$.

Note that in this simulation of CFSM-style communications, the counter is forcibly set to the (bounded) value corresponding to the message being exchanged, even for endpoints with type $\bullet$. We show, in the next section, another simulation of CFSM-style communications where one of the two peers is able to retain the value of its counter.

3 A Characterization of Topologies with Solvable Reachability

We investigate the power of systems of communicating one-counter machines with regard to their communication topology. Therefore, we introduce the reachability problem parametrized by a given topology. Recall that a full run of $[\![\mathcal{S}]\!]$ is a run from an initial configuration to a final configuration.

▶ **Definition 3.1.** Given a topology $\mathcal{T}$, the *reachability problem* for systems of communicating one-counter machines with topology $\mathcal{T}$, denoted by Rp-Sc1cm($\mathcal{T}$), is defined as follows:
Input: a system of communicating one-counter machines $\mathcal{S}$ with topology $\mathcal{T}$,
Output: whether there exists a full run in $[\![\mathcal{S}]\!]$.

The main result of the paper is a complete classification of the topologies that have a solvable reachability problem. We observe that, in absence of shunts, systems of communicating one-counter machines are still more expressive than Cfsm, but their reachability problems are decidable for the same topologies, namely, cycle-free topologies [17].

▶ **Theorem 3.2.** *Given a topology $\mathcal{T}$,* Rp-Sc1cm($\mathcal{T}$) *is decidable if and only if $\mathcal{T}$ is cycle-free and shunt-free.*

The proof of the theorem is presented at the end of this section for the "only if" direction, and in Section 4 for the "if" direction. Before that, let us provide a decomposition of topologies that are cycle-free and shunt-free. Observe that a weakly-connected topology is cycle-free if and only if there is a unique simple undirected path between every two processes.

▶ **Proposition 3.3.** *Let $\mathcal{T}$ be a weakly-connected topology with at least two processes. If $\mathcal{T}$ is cycle-free and shunt-free, then there are two distinct processes r, r', with $r \overset{c}{\leftrightarrow} r'$ for some channel c, such that, for every simple undirected path $(p_0, c_1, p_1, \ldots, c_n, p_n, d, q)$ with $p_0 \in \{r, r'\}$ and $q \notin \{r, r'\}$, the process q has type $\circ$ on the channel d.*

An example illustrating the proposition is provided in Figure 2(a). This weakly-connected topology is cycle-free and shunt-free. Therefore, its underlying undirected graph is a tree. The processes r and r' may be seen as two "roots", connected by a channel. All other processes are descendants of these two "roots", and have type $\circ$ on the channel (input or output) that leads to the root, as required by Proposition 3.3. Note, however, that r and r' are allowed to have type $\bullet$ on all channels. Recall that a process with type $\circ$ on a channel "loses" the value of its counter when it communicates over this channel. On the contrary, no loss of information occurs with type $\bullet$. But an endpoint with type $\bullet$ can simulate an endpoint with type $\circ$, by artificially "losing" the value of the local counter. We formalize this property by introducing the partial order $\sqsubseteq$ on $\{\circ, \bullet\}$ defined by $\circ \sqsubset \bullet$. This partial order is extended to endpoints in the natural way: $(p, *) \sqsubseteq (p', *')$ if $p = p'$ and $* \sqsubseteq *'$. Given two topologies $\mathcal{T} = \langle P_\mathcal{T}, C_\mathcal{T}, \mathsf{src}_\mathcal{T}, \mathsf{dst}_\mathcal{T} \rangle$ and $\mathcal{U} = \langle P_\mathcal{U}, C_\mathcal{U}, \mathsf{src}_\mathcal{U}, \mathsf{dst}_\mathcal{U} \rangle$, we say that $\mathcal{U}$ is a *sub-topology* of $\mathcal{T}$ if $P_\mathcal{U} \subseteq P_\mathcal{T}$, $C_\mathcal{U} \subseteq C_\mathcal{T}$, and, for every channel $c \in C_\mathcal{U}$, it holds that $\mathsf{src}_\mathcal{U}(c) \sqsubseteq \mathsf{src}_\mathcal{T}(c)$ and $\mathsf{dst}_\mathcal{U}(c) \sqsubseteq \mathsf{dst}_\mathcal{T}(c)$. As one would expect, sub-topologies have an easier reachability problem.

▶ **Proposition 3.4.** *For every topology $\mathcal{T}$ and for every sub-topology $\mathcal{U}$ of $\mathcal{T}$,* Rp-Sc1cm($\mathcal{U}$) *is reducible to* Rp-Sc1cm($\mathcal{T}$).

Cycle-freeness and Shunt-freeness of Decidable Topologies. In the remainder of this section, we prove the "only if" direction of Theorem 3.2, namely that Rp-Sc1cm($\mathcal{T}$) is undecidable if $\mathcal{T}$ contains a simple undirected cycle or a simple undirected shunt. As seen in Section 2, systems of communicating one-counter machines can simulate Cfsm, and

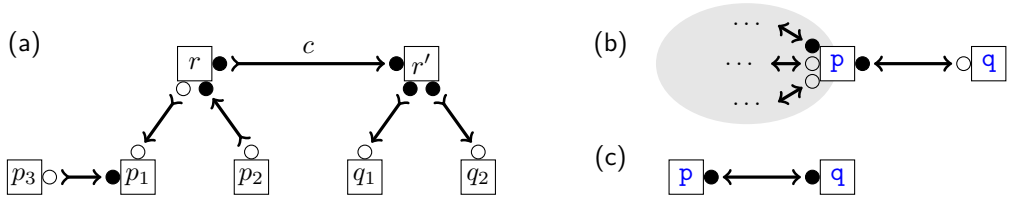

Figure 2 Topologies: (a) weakly connected cycle-free and shunt-free topology, (b) topology containing a leaf process q with type ∘ on its pendant channel, (c) decidable two-processes case.

the simulation preserves the topology. Moreover, the reachability problem for CFSM with topology $\mathcal{T}$ is known to be undecidable if $\mathcal{T}$ contains a simple undirected cycle [17, 14]. It follows that RP-SC1CM($\mathcal{T}$) is undecidable if $\mathcal{T}$ contains a simple undirected cycle. The following lemma completes the proof of the "only if" direction of Theorem 3.2.

▶ **Lemma 3.5.** *For every topology $\mathcal{T}$ containing a simple undirected shunt,* RP-SC1CM($\mathcal{T}$) *is undecidable.*

We explain the main ideas of the proof on the topology $\mathtt{p} \overset{\mathtt{c}}{\rightarrowtail} \mathtt{r} \overset{\mathtt{d}}{\leftarrowtail} \mathtt{q}$ where r has type ∘ on channels c and d, p has type • on c and q has type • on d. Let us call this topology $\mathcal{T}$. Notice that (p, c, r, d, q) is a simple undirected shunt. We show that the reachability problem for two-counters (Minsky) machines, which is known to be undecidable [16], is reducible to RP-SC1CM($\mathcal{T}$). Given a two-counters machine $\mathcal{M}$, one counter, say x, is maintained by p, and the other, say y, is maintained by q. Both processes p and q run a copy of $\mathcal{M}$, but they internalize (as add(0) actions) the counter actions of $\mathcal{M}$ that do not involve their counter. We only need to make sure that p and q take the same control path of $\mathcal{M}$. To this end, p and q send to r the transition rules that they traverse, and r checks that these rules are the same. However, p and q must not lose the value of their counter when communicating with r. So the simulation of CFSM presented in Section 2 cannot be used. Instead, p and q encode the transition rules within the counter value itself, send it to r, and let r decode and check this information.

Assume that $\mathcal{M}$ contains $K > 0$ transition rules, encoded as $0, \ldots, K-1$. Instead of storing the values x and y of x and y in their local counters, p and q store $K \cdot x$ and $K \cdot y$, respectively. So, increments and decrements in $\mathcal{M}$ are multiplied by the constant K in p and q. On the sender side, when p or q takes a transition rule encoded by $\delta \in \{0, \ldots, K-1\}$, it increments its counter by δ, sends it to r, and decrements its counter by δ to restore its value. On the receiver side, when r performs a c ? action, its counter is set to the message $m = \delta + (K \cdot x)$ sent by p, and r extracts the transition rule δ by computing (m mod K). The transition rules taken by q are decoded by r similarly.

The simulation guarantees that the two-counters machine has a full run if and only if the constructed system of communicating one-counter machines, with topology $\mathcal{T}$, has a full run. It follows that RP-SC1CM($\mathcal{T}$) is undecidable. Note that, by Proposition 3.4, the reachability problem RP-SC1CM($\mathcal{T}$) would also be undecidable (and even more so) if r had type • instead of ∘ on its output channels.

▶ Remark. We need at least one intermediary process r between p and q, to decode and check their messages. Indeed, direct communications between p and q would synchronize their local counters, thus making it impossible to maintain two counters.

4 Decidability of Cycle-free and Shunt-free Topologies

This section is devoted to the proof of the "if" direction of Theorem 3.2, namely that RP-SC1CM($\mathcal{T}$) is decidable if $\mathcal{T}$ is cycle-free and shunt-free. Without loss generality, we only consider weakly-connected topologies. The proof comprises three independent parts. Firstly, we provide a characterization, in terms of Presburger predicates, of reachability relations of one-counter machines. Secondly, we show that any leaf process with type $\circ$ on its pendant channel may be merged into its parent, thereby reducing the size of the topology. Iterating this reduction leads to a topology with only two processes and one channel. We show, in the third part, that RP-SC1CM($\mathcal{T}$) is decidable for such topologies.

Counter reachability relations of one-counter machines. A *one-counter machine* is a communicating one-counter machine $\mathcal{M} = \langle S, I, F, A, \Delta \rangle$ with no communication action, i.e., $A \subseteq A_{\text{cnt}}$. To fit our framework, we identify $\mathcal{M}$ with the system $\langle \mathcal{U}, (\mathcal{M}^p)_{p \in \{\mathtt{p}\}} \rangle$ of communicating one-counter machines, where $\mathcal{U} = \langle \{\mathtt{p}\}, \emptyset, \mathsf{src}, \mathsf{dst} \rangle$ is the topology with a single process $\mathtt{p}$ and no channel. We let RP-1CM denote the reachability problem for one-counter machines, formally RP-1CM = RP-SC1CM($\mathcal{U}$). It is well-known that RP-1CM is decidable since reachability is decidable for the more general class of pushdown systems.

In the next subsections, we show that, under certain conditions, two processes can be merged in a single "product" process (with only one counter). To do so, we summarize the behavior of a process between each communication action. This subsection is devoted to the characterization and computation of these summaries.

Let $\mathcal{M} = \langle S, I, F, A, \Delta \rangle$ be a one-counter machine. The *counter reachability relation* of $\mathcal{M}$ is the set of all pairs $(x, y) \in \mathbb{N} \times \mathbb{N}$ such that, for some $s \in I$ and $t \in F$, there exists a run from (s, x) to (t, y). To characterize counter reachability relations, we introduce the following class of binary Presburger predicates. We consider two distinguished Presburger variables $\mathtt{x}$ and $\mathtt{y}$. In short, one-counter Presburger predicates can express properties of $\mathtt{x}$, of $\mathtt{y}$, and of their differences $\mathtt{x} - \mathtt{y}$ and $\mathtt{y} - \mathtt{x}$. Formally, the class of *one-counter Presburger predicates* is generated by the grammar:

$$\psi \quad ::= \quad \varphi(\mathtt{x}) \mid \varphi(\mathtt{y}) \mid \exists z \cdot (\mathtt{x} = \mathtt{y} + z \wedge \varphi(z)) \mid \exists z \cdot (\mathtt{y} = \mathtt{x} + z \wedge \varphi(z)) \mid \psi \wedge \psi \mid \psi \vee \psi \mid \mathtt{tt} \mid \mathtt{ff}$$

where φ ranges over the set $\mathcal{P}_1$ of unary Presburger predicates. The binary relation *defined* by a one-counter Presburger predicate ψ is the set of all pairs $(x, y) \in \mathbb{N} \times \mathbb{N}$ such that the valuation $\{\mathtt{x} \mapsto x, \mathtt{y} \mapsto y\}$ satisfies ψ.

We first show that counter reachability relations are definable by one-counter Presburger predicates, for the class of one-counter machines with zero-tests only. Formally, a one-counter machine $\mathcal{M} = \langle S, I, F, A, \Delta \rangle$ is called *basic* if $A \subseteq \{\mathtt{add}(k) \mid k \in \mathbb{Z}\} \cup \{\mathtt{test}(\mathtt{x} = 0)\}$.

▶ **Lemma 4.1.** *For every basic one-counter machine $\mathcal{M}$, the counter reachability relation of $\mathcal{M}$ is defined by a one-counter Presburger predicate.*

However, the converse of the lemma does not hold. Consider, for instance, the one-counter Presburger predicate $\psi = \exists \mathtt{k} \cdot (\mathtt{x} = \mathtt{k} + \mathtt{k}) \wedge (\mathtt{x} = \mathtt{y})$. In a basic one-counter machine, it is not possible to check that a given, a priori unknown value $\mathtt{x}$ is even without "losing" this value. We need the additional expressive power stemming from Presburger tests.

We now show that counter reachability relations (of arbitrary one-counter machines) are precisely the relations definable by one-counter Presburger predicates. This entails, in particular, that counter reachability relations are closed under intersection. We will use this

property in the proof of Lemma 4.4. On the logical side, we obtain that the class of relations definable by one-counter Presburger predicates is closed under transitive closure.

▶ **Theorem 4.2.** *For every binary relation $R \subseteq \mathbb{N} \times \mathbb{N}$, the two following assertions are equivalent:*

- *R is the counter reachability relation of a one-counter machine,*
- *R is defined by a one-counter Presburger predicate.*

▶ Remark. The proof of Theorem 4.2 is constructive, in the sense that a one-counter Presburger predicate is computable from a given one-counter machine, and vice versa.

Merging leaf processes. We show how to reduce the number of processes in a system of communicating one-counter machines, by merging a leaf process with type $\circ$ on its pendant channel into its parent. Let $\mathcal{U} = \langle P_{\mathcal{U}}, C_{\mathcal{U}}, \mathsf{src}_{\mathcal{U}}, \mathsf{dst}_{\mathcal{U}} \rangle$ be a topology, and select a distinguished process $\mathtt{p}$ in $P_{\mathcal{U}}$. We add to the topology a new process $\mathtt{q} \notin P_{\mathcal{U}}$ and a new channel $\mathtt{c} \notin C_{\mathcal{U}}$ between $\mathtt{p}$ and $\mathtt{q}$. Formally, we consider any topology $\mathcal{T} = \langle P, C, \mathsf{src}, \mathsf{dst} \rangle$ with set of processes $P = P_{\mathcal{U}} \cup \{\mathtt{q}\}$ and set of channels $C = C_{\mathcal{U}} \cup \{\mathtt{c}\}$, whose source and destination mappings coincide with those of $\mathcal{U}$ on C, and such that $\mathtt{p} \overset{\mathtt{c}}{\leftrightarrow} \mathtt{q}$. Observe that $C(\mathtt{q}) = \{\mathtt{c}\}$, hence, $\mathtt{q}$ is a leaf process with pendant channel $\mathtt{c}$. The topology $\mathcal{T}$ is depicted on Figure 2(b).

▶ **Lemma 4.3.** *If $\mathtt{p}$ has type $\bullet$ on $\mathtt{c}$ and $\mathtt{q}$ has type $\circ$ on $\mathtt{c}$ then* Rp-Sc1cm$(\mathcal{T})$ *is reducible to* Rp-Sc1cm$(\mathcal{U})$.

Let us explain the main ideas of the proof. Assume that $\mathtt{c}$ is directed as $\mathtt{p} \overset{\mathtt{c}}{\rightarrowtail} \mathtt{q}$. Consider a system of communicating one-counter machines $\mathcal{S} = \langle \mathcal{T}, (\mathcal{M}^p)_{p \in P} \rangle$. To simulate $\mathcal{S}$ over the topology $\mathcal{U}$, we merge processes $\mathtt{p}$ and $\mathtt{q}$ in a single "product" process $\widehat{\mathtt{p}}$. So, the communicating one-counter machines $\mathcal{M}^p$ are kept unchanged for all processes in $p \in P \setminus \{\mathtt{p}, \mathtt{q}\}$. But the process $\widehat{\mathtt{p}}$ must simulate both processes $\mathtt{p}$ and $\mathtt{q}$, as well as the channel $\mathtt{c}$ in-between. We choose a specific interleaving of $\mathtt{p}$ and $\mathtt{q}$ where $\mathtt{c}$ is almost always empty, and such that $\widehat{\mathtt{p}}$, which has a single counter, is able to retain both $\mathtt{p}$'s counter and $\mathtt{q}$'s counter.

In essence, $\widehat{\mathtt{p}}$ behaves as $\mathtt{p}$, but also maintains, in its state, the local state of $\mathtt{q}$ as well as an abstraction of $\mathtt{q}$'s counter. We abstract $\mathtt{q}$'s counter by the set $\{0, \bot, =\}$, where 0 means zero, $\bot$ means some unknown value, and $=$ means that $\mathtt{q}$'s counter holds the same value as $\mathtt{p}$'s counter. Furthermore, the process $\mathtt{q}$ is always scheduled first. Since $\mathtt{c}$ is the only channel with source or destination $\mathtt{q}$, this means, in particular, that every reception by $\mathtt{q}$ from $\mathtt{c}$ occurs immediately after the matching emission by $\mathtt{p}$ on $\mathtt{c}$. When $\widehat{\mathtt{p}}$ simulates an emission by $\mathtt{p}$ on $\mathtt{c}$ and the matching reception by $\mathtt{q}$, it internalizes this synchronization $\mathtt{c}! \cdot \mathtt{c}?$, and sets $\mathtt{q}$'s abstract counter to $=$. Indeed, since $\mathtt{q}$ has type $\circ$ on $\mathtt{c}$, the reception by $\mathtt{q}$ from $\mathtt{c}$ overwrites its counter with the value of $\mathtt{p}$'s counter. Then, $\widehat{\mathtt{p}}$ simulates, in one step, the behavior of $\mathtt{q}$ from this matching reception to the next reception. Observe that the next reception of $\mathtt{q}$ from $\mathtt{c}$ will, again, overwrite its counter. Therefore, thanks to Theorem 4.2, this behavior of $\mathtt{q}$ can be summarized in a single Presburger test, that accounts for the local state reached by $\mathtt{q}$. This way, $\widehat{\mathtt{p}}$ does not need to maintain the value held by $\mathtt{q}$'s counter. The construction guarantees that $\mathcal{S}$ has a full run if and only if the resulting system of communicating one-counter machines, with topology $\mathcal{U}$, has a full run.

The proof for the other direction $\mathtt{q} \overset{\mathtt{c}}{\rightarrowtail} \mathtt{p}$ is similar. However, instead of scheduling $\mathtt{q}$ first, it is now scheduled last.

Two processes connected by one channel. We now consider the topology depicted on Figure 2(c), with two distinct processes $\mathtt{p}$ and $\mathtt{q}$ and a channel from $\mathtt{p}$ to $\mathtt{q}$ with type $\bullet$ on both endpoints. Formally, $\mathcal{T} = \langle \{\mathtt{p}, \mathtt{q}\}, \{\mathtt{c}\}, \mathsf{src}, \mathsf{dst} \rangle$ with $\mathsf{src}(\mathtt{c}) = (\mathtt{p}, \bullet)$ and $\mathsf{dst}(\mathtt{c}) = (\mathtt{q}, \bullet)$.

▶ **Lemma 4.4.** RP-SC1CM($\mathcal{T}$) *is reducible to* RP-1CM.

Informally, given a system of communicating one-counter machines $\mathcal{S} = \langle \mathcal{T}, (\mathcal{M}^p)_{p \in P} \rangle$, we construct a one-counter machine $\mathcal{N}$ that simulates the "product" of p and q. As in the proof of Lemma 4.3, we schedule the sender last (here, p) and the receiver first (here, q). Thus, emissions c! and receptions c? occur consecutively, with no other action in between. Since p and q have type $\bullet$ on c, each sequence of actions c! · c? may occur only if p's counter and q's counter hold the same value. So $\mathcal{N}$ internalizes each synchronization c! · c?, and simulates, in one step, the behavior of p and q from one synchronization to the next. This is possible thanks to Theorem 4.2, which entails that counter reachability relations are (effectively) closed under intersection. The construction guarantees that $\mathcal{S}$ has a full run if and only if the constructed one-counter machine $\mathcal{N}$ has a full run.

Wrap up. We now have the necessary ingredients to prove the "if" direction of Theorem 3.2. Consider a weakly-connected topology $\mathcal{T}$ that is both cycle-free and shunt-free. We show that RP-SC1CM($\mathcal{T}$) is reducible to RP-1CM. If $\mathcal{T}$ contains only one process, then $\mathcal{T}$ contains no channel as it is cycle-free, hence, RP-SC1CM($\mathcal{T}$) is obviously reducible to RP-1CM. Assume that $\mathcal{T}$ contains at least two processes. By Proposition 3.3, there exists two distinct processes r, r' and a channel c, with $r \overset{c}{\leftrightarrow} r'$, such that, for every simple undirected path $(p_0, c_1, p_1, \ldots, c_n, p_n, d, q)$ with $p_0 \in \{r, r'\}$ and $q \notin \{r, r'\}$, the process q has type $\circ$ on the channel d. Moreover, according to Proposition 3.4, we may replace some endpoints $(p, \circ)$ by $(p, \bullet)$, as the reachability problem RP-SC1CM($\mathcal{T}$) is reducible to the reachability problem for the transformed topology. So we assume, without loss of generality, that for every simple undirected path $(p_0, c_1, p_1, \ldots, c_n, p_n, p, d, q)$ with $p_0 \in \{r, r'\}$, the process p has type $\bullet$ on the channel d. In particular, r and r' have type $\bullet$ on c.

Since $\mathcal{T}$ is cycle-free, its underlying undirected graph $(P, \{\overset{c}{\leftrightarrow}\}_{c \in C})$ is a tree. Pick a leaf process q that is distinct from r and r' (if any). Let $\mathcal{T} - q$ denote the topology obtained from $\mathcal{T}$ by removing the process q as well as its pendant channel. The simple undirected path from r to q ends with a channel $p \overset{d}{\leftrightarrow} q$ that satisfies $C(q) = \{d\}$, p has type $\bullet$ on d and q has type $\circ$ on d. It follows from Lemma 4.3 that RP-SC1CM($\mathcal{T}$) is reducible to RP-SC1CM($\mathcal{T} - q$). By iterating this elimination technique in a bottom-up fashion, we obtain that RP-SC1CM($\mathcal{T}$) is reducible to RP-SC1CM($\mathcal{U}$) where $\mathcal{U}$ is the topology consisting of the two processes r, r' and the single channel c. According to Lemma 4.4, RP-SC1CM($\mathcal{U}$) is reducible to RP-1CM. We conclude that RP-SC1CM($\mathcal{T}$) is reducible to RP-1CM. Since the latter decidable, we get that the former is decidable, too.

5 Systems with Eager Communication

As seen in our motivating example of Figure 1, cyclic topologies are the backbone of communication protocols. However, already for CFSM, the reachability problem is undecidable in presence of cycles, which is also mirrored in Theorem 3.2. In this section, we consider a restriction to so-called eager runs. This restriction provides an under-approximative answer to the reachability problem RP-SC1CM($\mathcal{T}$) considered in the previous sections. Eager runs are close to globally 1-bounded runs, and have been successfully applied, in combination with other restrictions, to the reachability analysis of communicating pushdown processes [13].

▶ **Definition 5.1.** A full run $\rho = (\sigma_0, a_1, \sigma_1, \ldots, a_n, \sigma_n)$ in $[\![\mathcal{S}]\!]$ is called *eager* if, for every channel c and for every index $i \in \{1, \ldots, n-1\}$, it holds that $a_i = c!$ if and only if $a_{i+1} = c?$.

Thus, eagerness transforms asynchronous message-passing communications into rendezvous synchronizations. This may seem rather restrictive. Actually, eagerness is equivalent, up to re-ordering[1], to the requirement that all other channels be empty when one channel is transferring a message [13]. Therefore, eagerness encompasses half-duplex communication.

The *eager-reachability problem* RP-SC1CM-EAGER($\mathcal{T}$) is defined in the same way as RP-SC1CM($\mathcal{T}$) except that we search for a full run that must be eager. By definition, this problem provides an under-approximative answer to RP-SC1CM($\mathcal{T}$). This under-approximation is exact when the topology is cycle-free. Indeed, for such topologies, full runs can be re-ordered into eager ones [13]. It follows from Theorem 3.2 that, for every cycle-free topology $\mathcal{T}$, RP-SC1CM-EAGER($\mathcal{T}$) is decidable if and only if $\mathcal{T}$ is shunt-free. Hence, eagerness is only interesting in presence of cycles. For the remainder of this section, we focus on cyclic communication. The following proposition establishes the decidability frontier of the eager-reachability problem for the particular case of strongly-connected topologies.

▶ **Proposition 5.2.** *Given a strongly-connected topology $\mathcal{T}$,* RP-SC1CM-EAGER($\mathcal{T}$) *is decidable if and only if $\mathcal{T}$ contains at most two processes.*

We first consider the simplest strongly-connected topology with two processes $\mathtt{p} \overset{\mathtt{c}}{\rightarrowtail} \mathtt{q} \overset{\mathtt{d}}{\rightarrowtail} \mathtt{p}$, where all channel endpoints have type $\bullet$. Then, eagerness allows us to reverse the direction of a channel, leading to $\mathtt{p} \overset{\mathtt{c}}{\rightarrowtail} \mathtt{q} \overset{\mathtt{d}}{\leftarrowtail} \mathtt{p}$. With the same encoding as in Lemma 3.5, we may tag each message by the channel c or d that it is sent over. As eager message passing only uses one channel at a time, we can assert that all messages are now passed over one common channel. Hence we can apply the decidability result of Lemma 4.4 on two processes connected by one channel. This construction can be extended to more than two channels between p and q. A strongly-connected topology may also contain self-loops, but they become irrelevant by the restriction to eager runs. Finally, we extend this result to topologies with channel endpoints of type $\circ$ by Proposition 3.4 (generalized to eager-reachability).

For the converse, consider a strongly-connected component with at least three processes. We may assume, without loss generality, that all channel endpoints have type $\circ$. The component necessarily contains (a) a directed cycle of length at least three, i.e., assuming for simplicity that the length is three, a sub-topology $\mathcal{T}_a$ of the form $\mathtt{p} \overset{\mathtt{c}}{\rightarrowtail} \mathtt{q} \overset{\mathtt{d}}{\rightarrowtail} \mathtt{r} \overset{\mathtt{e}}{\rightarrowtail} \mathtt{p}$, or ($b$) two directed cycles, each of length two, that are disjoint except for one common process, i.e., a sub-topology $\mathcal{T}_b$ of the form $\mathtt{q} \overset{\mathtt{c}}{\rightarrowtail} \mathtt{p} \overset{\mathtt{d}}{\rightarrowtail} \mathtt{r} \overset{\mathtt{e}}{\rightarrowtail} \mathtt{p} \overset{\mathtt{f}}{\rightarrowtail} \mathtt{q}$. We show a reduction from the reachability problem for two-counters machines. The restriction to eager runs guarantees that each send is immediately followed by the matching receive. We use this restriction to implement a protocol that gives one distinguished process access to the two counters, the latter being stored and passed around in the topology without getting lost. In the case of $\mathcal{T}_a$, process p simulates the two-counters machine by maintaining one of the counters locally, and the other at r. To let p use the other counter, the protocol ensures that we switch the counters by using q as buffer. In the case of $\mathcal{T}_b$, the two-counters machine is simulated by p, while q and r are used as registers for either one of the two counters.

Let us come back to the sliding window protocol of Figure 1. Assume that, in both processes, receptions have precedence over transmissions. This priority ensures that channels are used in a half-duplex way. By [13], every full run can then be re-ordered into an eager one. Since the topology of Figure 1 falls in the scope of the previous proposition, we can decide whether the protocol is safe or not (when priority is given to receptions).

[1] A run ρ can be *re-ordered* into a run ρ' if ρ can be transformed into ρ' by iteratively commuting adjacent transitions that (i) are from different processes, and (ii) do *not* form a matching send/receive pair.

6 Conclusion and Perspectives

Systems of communicating one-counter machines introduce two additional sources of infinity with respect to CFSM, namely, the infinite message alphabet and the local counters. Thanks to a characterization of one-counter reachability relations in terms of binary Presburger predicates, we have obtained a complete classification of the topologies having a solvable reachability question. This shows, in particular, that decidable topologies are the same as for the weaker model of CFSM (provided that they contain no shunt). To address topologies allowing mutual communications, we have considered an under-approximative approach by restricting runs to eager ones. As a preliminary result, we have characterized the strongly-connected topologies that have a solvable eager-reachability question. A complete characterization of decidable topologies for eager reachability is currently under investigation. Further, we plan to extend our results from counters to stacks, i.e., to systems of communicating pushdown machines that can exchange the value of their stacks.

References

1 P. Abdulla, B. Jonsson. Verifying programs with unreliable channels. *Information and Computation*, 127(2):91–101, 1996.
2 S. Böhm, S. Göller, P. Jančar. Bisimilarity of one-counter processes is PSPACE-complete. In *Proc. CONCUR'10*, *LNCS* 6269, pp. 177–191. Springer, 2010.
3 M. Bojańczyk, C. David, A. Muscholl, T. Schwentick, L. Segoufin. Two-variable logic on data words. *ACM Trans. Computational Logic*, 12(4):27, 2011.
4 B. Bollig, A. Cyriac, P. Gastin, K. Narayan Kumar. Model checking languages of data words. In *Proc. FOSSACSS'12*, *LNCS* 7213, pp. 391–405. Springer, 2012.
5 A. Bouajjani, P. Habermehl, R. Mayr. Automatic verification of recursive procedures with one integer parameter. *Theoretical Computer Science*, 295:85–106, 2003.
6 D. Brand, P. Zafiropoulo. On communicating finite-state machines. Research Report 1053, IBM Zürich Research Laboratory, 1981.
7 P. Chambart, P. Schnoebelen. Mixing lossy and perfect fifo channels. In *Proc. CONCUR'08*, *LNCS* 5201, pp. 340–355, 2008.
8 S. Demri, R. Lazic, A. Sangnier. Model checking freeze LTL over one-counter automata. In *Proc. FOSSACS'08*, *LNCS* 4962, pp. 490–504. Springer, 2008.
9 A. Finkel. Decidability of the termination problem for completely specified protocols. *Distributed Computing*, 7(3):129–135, 1994.
10 A. Finkel, G. Memmi. Fifo nets: a new model of parallel computation. In *Proc. TCS'83*, *LNCS* 145, pp. 111–121. Springer, 1983.
11 A. Finkel, G. Sutre. Decidability of reachability problems for classes of two counters automata. In *Proc. STACS'00*, *LNCS* 1770, pp. 346–357. Springer, 2000.
12 S. Göller, C. Haase, J. Ouaknine, J. Worrell. Branching-time model checking of parametric one-counter automata. In *Proc. FOSSACS'12*, *LNCS* 7213, pp. 406–420. Springer, 2012.
13 A. Heußner, J. Leroux, A. Muscholl, G. Sutre. Reachability analysis of communicating pushdown systems. *Logical Methods in Computer Science*, 8(3:23):1–20, 2012.
14 S. La Torre, P. Madhusudan, G. Parlato. Context-bounded analysis of concurrent queue systems. In *Proc. TACAS'08*, *LNCS* 4963, pp. 299–314. Springer, 2008.
15 T. Le Gall, B. Jeannet. Lattice automata In *Proc. SAS'07*, *LNCS* 4634, pp. 52–68. Springer, 2007.
16 M. Minsky. *Computation: Finite and Infinite Machines*. Prentice-Hall, 1967.
17 J.K. Pachl. Reachability problems for communicating finite state machines. Research Report CS-82-11, Dept. of C.S. Univ. of Waterloo, 1982.